TRANSIMS and the Hierarchical Data Format

B. W. Bush
Los Alamos National Laboratory
12 June 1997

Abstract

The Hierarchical Data Format (HDF) is a general-purposed scientific data format developed at the National Center for Supercomputing Applications. It supports metadata, compression, and a variety of data structures (multidimensional arrays, raster images, tables). FORTRAN 77 and ANSI C programming interfaces are available for it and a wide variety of visualization tools read HDF files. We discuss the features of this file format and its possible uses in TRANSIMS.

Outline

- *HDF**
  - *motivation*
  - *features*
  - *platforms*
  - *application program interfaces*
  - *limitations*
  - *analysis software*
- *TRANSIMS*
  - *storage requirements*
  - *possible uses of HDF*
  - *future plans*

**Most of the material presented in this section has been extracted verbatim from HDF documentation.*

Overview

- *HDF stands for Hierarchical Data Format.*
- *HDF is a library and multi-object file format for the transfer of graphical and numerical data between machines.*
- *HDF is a library and platform independent data format for the storage and exchange of scientific data.*
- *It includes Fortran and C calling interfaces, and utilities for analyzing and converting HDF data files.*
- *HDF is developed and supported by NCSA, and is available in the public domain.*
- *HDF is used world-wide in many fields, including Environmental Science, Neutron Scattering, Non-Destructive Testing, and Aerospace, to name a few.*
- *Scientific projects that use HDF include NASA's Mission to Planet Earth, and the DOE's Accelerated Strategic Computing Initiative.*

Motivation

- *Scientists commonly generate and process data files on several different machines, use various software packages to process files and share data files with others who use different machines and software.*
- *Also, they may include different kinds of information within one particular file, or within a group of files, and the mixture of these different kinds of information may vary from one file to another.*
- *Files may be conceptually related but physically separated.*
- *It is also possible that data may be related only in the scientist's conception of the data; no physical relationship may exist.*
- *HDF addresses these problems by providing a general-purpose file structure.*

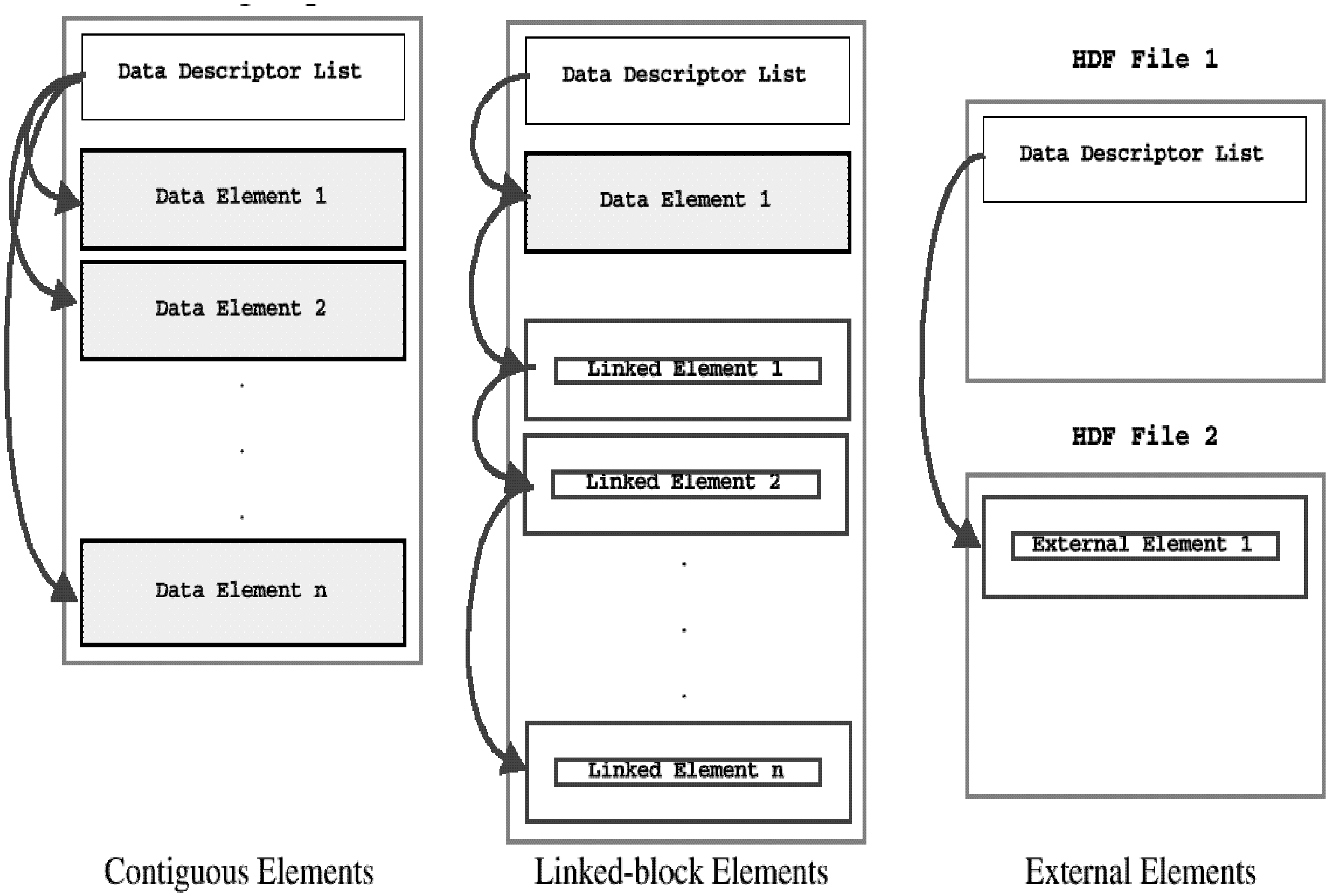
Features

- *versatile*
  - *HDF supports several different data models.*
  - *Each data model defines a specific aggregate data type and provides an API for reading, writing, and organizing data and metadata of the corresponding type.*
  - *Data models supported include multidimensional arrays, raster images, and tables.*
- *self-describing*
  - *An application is able to interpret the structure and contents of a file without any outside information.*
- *flexible*
  - *With HDF, you can mix and match related objects together in one file and then access them as a group or as individual objects.*
  - *Users can also create their own grouping structures.*

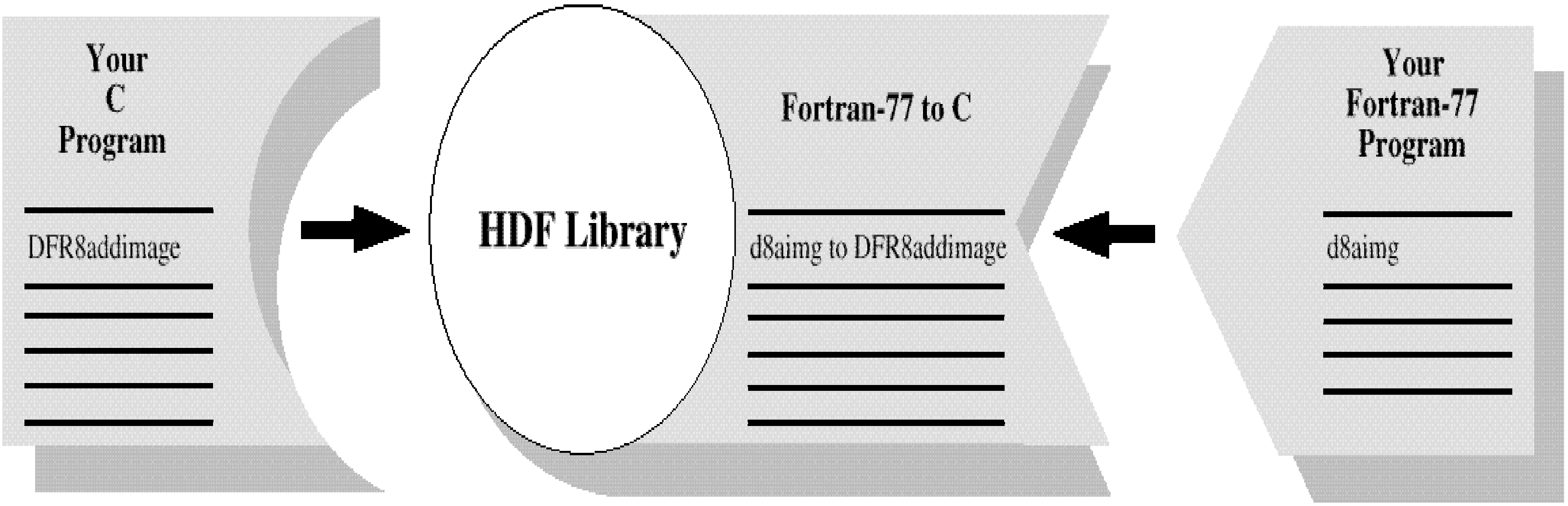
Features (continued)

- *extensible*
  - *It can easily accommodate new data models, regardless of whether they are added by the HDF development team or by HDF users.*
- *portable*
  - *HDF files can be shared across most common platforms, including many workstations and high performance computers.*
  - *An HDF file created on one computer can be read on a different system without modification.*

Data Storage Options



C and FORTRAN Language Interfaces



Interface Documentation Example

SDreadattr/sfrnatt/sfrcatt

intn SDreadattr(int32 [*file, sds, dim*]_*id*, int32 *attr_index*, VOIDP *data*)

[*file, sds, dim*]_*id*	IN:	Identifier of the object the attribute is to be attached to: an *file_id* for a file, an *sds_id* for an SDS or a *dim_id* for a dimension
attr_index	IN:	Index of the attribute to be read
data	OUT:	Buffer for the attribute values

Purpose Reads the values of an attribute.

Return value Returns `SUCCEED` (or `0`) if successful and `FAIL` (or `-1`) otherwise.

Description It's assumed that the user has called **SDattrinfo** and that the buffer is large enough to store the data. If an attribute has multiple values stored for it, this routine will return all of them. It is not possible to read a subset of attribute values.

Note that this routine has two Fortran-77 versions: **sfrnatt** and **sfrcatt**. The **sfrnatt** routine reads numeric attribute data and **sfrcatt** reads character attribute data.

The index returned as the *attr_index* argument is one-based.

FORTRAN

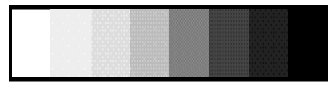
```
integer function sfrnatt([file, sds, dim]_id, attr_index, data)

integer [file, sds, dim]_id, attr_index
<valid numeric data> data


integer function sfrcatt([file, sds, dim]_id, attr_index, data)

integer [file, sds, dim]_id, attr_index
character* (*) data
```

Supported and Tested Platforms for Version 4.0

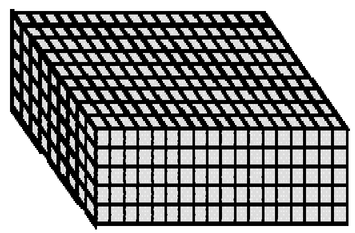
Platform Name	Basic HDF Library?	HDF/netCDF Library?
Sun Microsystems Sun4/SunOS V4.1.4	Yes	Yes
Sun Microsystems Sun4/Solaris V2.3	Yes	Yes
IBM RS6000/AIX	Yes	Yes
Silicon Graphics Indy/IRIX V5.3	Yes	Yes
Silicon Graphics Indy/IRIX V6.1.32	Yes	Yes
Silicon Graphics Indy/IRIX V6.1.64	Yes	Yes
Silicon Graphics Power Challenge/IRIX V6.1	Yes	Yes
Convex Exemplar/HPUX	Yes	Yes
Cray Y-MP/UNICOS	Yes	Yes
Cray C90/UNICOS	Yes	Yes
Thinking Machines CM5	Yes	Yes
Hewlett-Packard HP9000-735/HPUX V9.01	Yes	Yes
Intel x86/MS-DOS V6.0	Yes	No
Intel x86/MS-Windows V3.1	Yes	No
Intel x86/MS-Windows NT	Yes	Yes
Intel x86/Sun Microsystems Solarisx86*	Yes	Yes
Intel x86/Linux V1.2.4 (a.out Libraries)	Yes	Yes
Intel x86/Linux V1.2.4 (ELF Libraries)**	Yes	Yes
Intel x86/FreeBSD V2.0	Yes	Yes
DEC Alpha/OSF1 V3.0	Yes	Yes
Apple Macintosh 68000/MacOS V7.5**	No	No

Primary Data Structures

Raster Images
(8-bit, 24-bit and General Raster)



Palette



Scientific data
(Multi-dimensional arrays)

This HDF file contains one example of each HDF data type.

Annotation

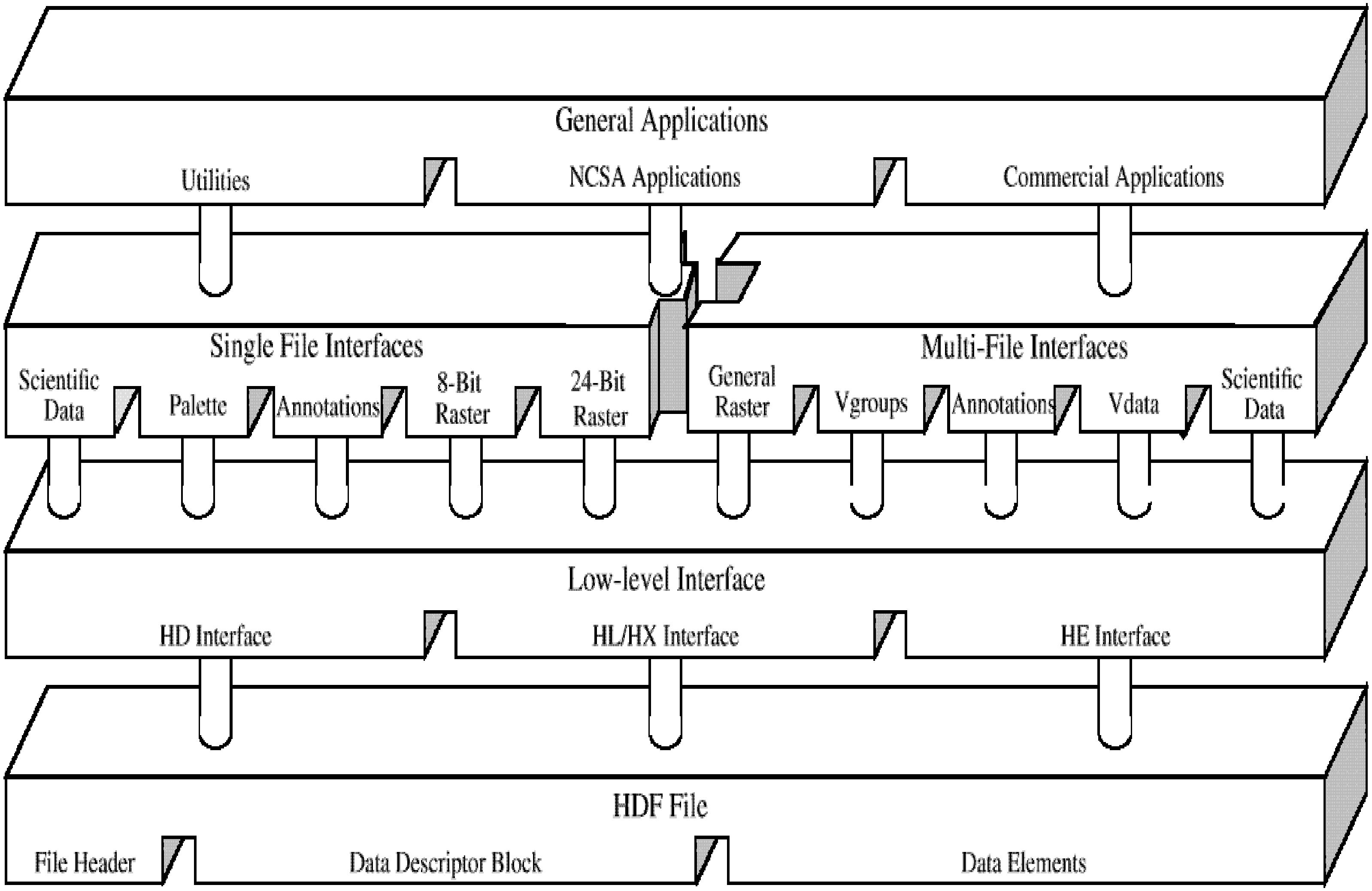
X	Y	Z
4.1586	25.697	.78341
6.9214	38.451	.77549
2.9182	67.904	.87401
4.0913	58.743	.90428
3.8510	21.048	.76306

Vdata
(Tables of ints, floats, and chars)

Application Program Interfaces

- ***DFR8*** *stores, manages and retrieves 8-bit raster images, their dimensions and palettes in one file.*
- ***DFP*** *stores and retrieves 8-bit palettes in one file.*
- ***DF24*** *stores, manages and retrieves 24-bit images and their dimensions in one file.*
- ***SD and DFSD*** *store, manage and retrieve multi-dimensional arrays of integer or floating-point data, along with their dimensions and attributes, in more than one file.*
- ***AN*** *stores, manages and retrieves text strings used to describe a file or any of the data elements it contains.*
- ***VS*** *stores, manages and retrieves multivariate data stored as records in a table.*
- ***V*** *creates groups of any primary HDF object type.*
- ***GR*** *stores, manages and retrieves raster images of several bit lengths, their dimensions and palettes in more than one file.*

Levels of Interaction



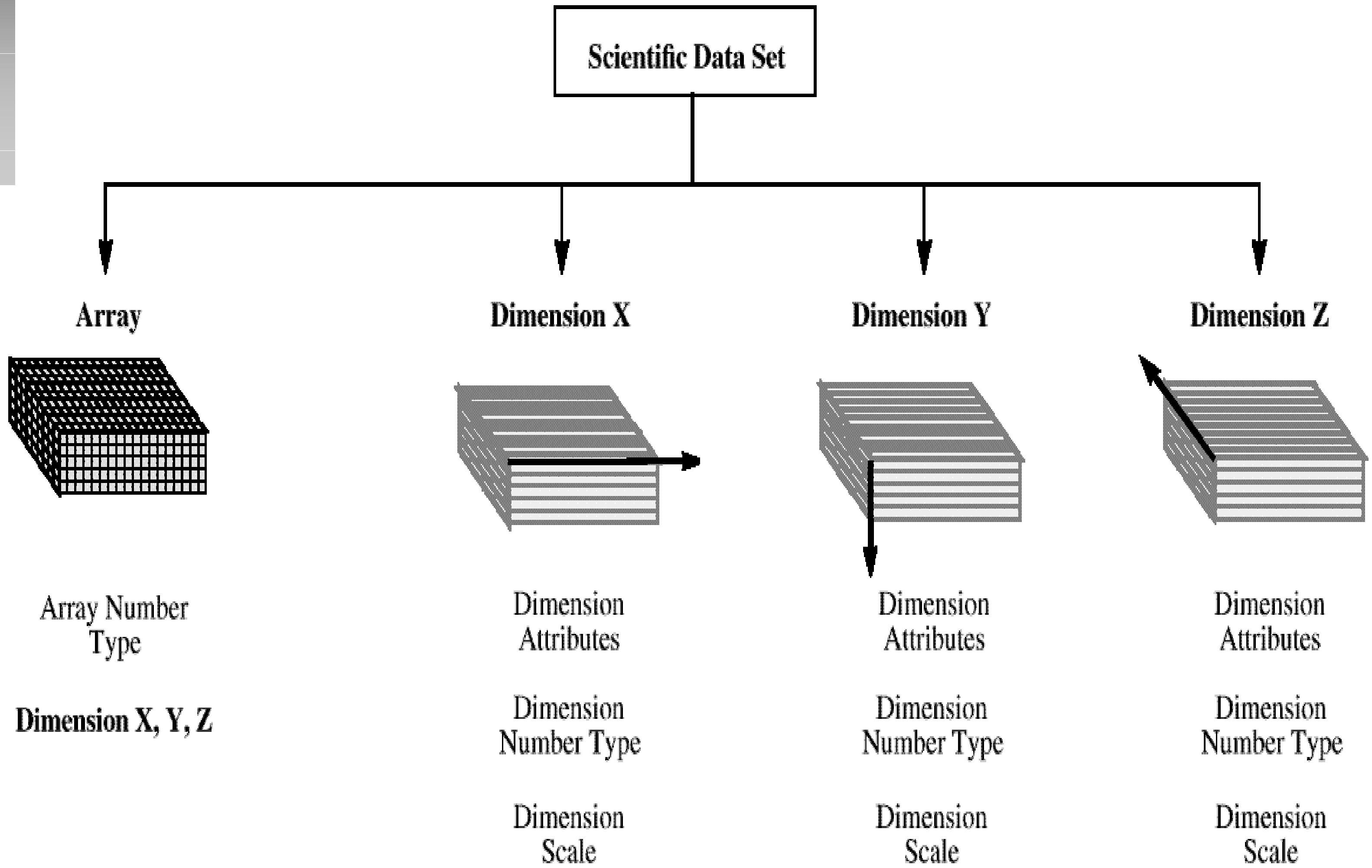
Data Type Definitions

Data Type	C	Fortran-77
8-bit signed integer	`int8`	Not supported.
8-bit unsigned integer	`uint8`	`character*1`
16-bit signed integer	`int16`	`integer*2`
16-bit unsigned integer	`uint16`	Not supported.
32-bit signed integer	`int32`	`integer*4`
32-bit unsigned integer	`uint32`	Not supported.
32-bit floating point number	`float32`	`real*4`
64-bit floating point number	`float64`	`real*8`

Scientific Data Sets (SD API)

- *In HDF, any multi-dimensional array qualifies as a* ***scientific data set*** *or* ***SDS*** *if it's associated with*
  - *a dimension record and*
  - *a data type.*
- *In addition to providing a framework for storing arrays of arbitrary dimensions and data type, the SDS data model supports*
  - *dimension scales,*
  - *user-defined attributes and*
  - *predefined attributes.*

Three-Dimensional SDS Example



SDS Program Example

C:

```c
#include "hdf.h"

#define X_LENGTH 5
#define Y_LENGTH 16

main( )
{

int32 sd_id, sds_id, retn;
int32 dims[2], start[2], edges[2], rank;
int16 array_data[Y_LENGTH][X_LENGTH];
intn i, j;

/* Create and open the file and initiate the SD interface. */
sd_id = SDstart("Example3.hdf", DFACC_CREATE);

/* Define the rank and dimensions of the data set to be created. */
rank = 2;
dims[0] = Y_LENGTH;
dims[1] = X_LENGTH;

/* Create the array data set. */
sds_id = SDcreate(sd_id, "Ex_array_3", DFNT_INT16, rank, dims);

/* Fill the stored-data array with values. */
for (j = 0; j < Y_LENGTH; j++)
    for (i = 0; i < X_LENGTH; i++)
        array_data[j][i] = (i + j) + 1;

/* Define the location, pattern, and size of the data set */
for (i = 0; i < rank; i++) {
    start[i] = 0;
    edges[i] = dims[i];
}

/* Write the stored data to the "Ex_Array_3" data set. The fifth \
* argument must be explicitly cast to a generic pointer to conform \
* to the HDF API definition for SDwritedata.*/
retn = SDwritedata(sds_id, start, NULL, edges, (VOIDP)array_data);

/* Terminate access to the array. */
retn = SDendaccess(sds_id);

/* Terminate access to the SD interface and close the file. */
retn = SDend(sd_id);

}
```

FORTRAN:

```fortran
      PROGRAM FILLED ARRAY

      integer*4 sd_id, sds_id, rank
      integer dims(2), start(2), edges(2), stride(2)
      integer i, j, retn
      integer sfstart, sfcreate, sfwdata, sfendacc, sfend

C     DFACC_CREATE and DFNT_INT16 are defined in hdf.h.
      integer*4 DFACC_CREATE, DFNT_INT16
      integer*4 X_LENGTH, Y_LENGTH
      parameter (DFACC_CREATE = 4, DFNT_INT16 = 22, X_LENGTH = 5,
     +           Y_LENGTH = 16)
      integer*2 array_data(X_LENGTH, Y_LENGTH)

C     Create and open the file and initiate the SD interface.
      sd_id = sfstart('Example3.hdf', DFACC_CREATE)

C     Define the rank and dimensions of the data set to be created.
      rank = 2
      dims(1) = X_LENGTH
      dims(2) = Y_LENGTH

C     Create the data set.
      sds_id = sfcreate(sd_id, 'Ex_array_3', DFNT_INT16, rank, dims)

C     Fill the stored-data array with values.
      do 20 j = 1, Y_LENGTH
          do 10 i = 1, X_LENGTH
              array_data(i, j) = i + j - 1
10        continue
20    continue

C     Define the location, pattern, and size of the data set
C     that will be written to.
      start(1) = 0
      start(2) = 0
      edges(1) = X_LENGTH
      edges(2) = Y_LENGTH
      stride(1) = 1
      stride(2) = 1

C     Write the stored data to the "Ex_array_3" data set.
      retn = sfwdata(sds_id, start, stride, edges, array_data)

C     Terminate access to the array.
      retn = sfendacc(sds_id)

C     Terminate access to the SD interface and close the file.
      retn = sfend(sd_id)

      end
```

SDS Interface Routines

Category	Routine Name		Description
	C	Fortran-77	
Access	SDend	sfend	Closes the file and clean up memory.
	SDendaccess	sfendacc	Disposes of a data set identifier, flush out metadata and order information.
	SDselect	sfselect	Returns the identifier of the specified data set.
	SDstart	sfstart	Initializes the SD interface.
Read and Write	SDcreate	sfcreate	Creates a new data set.
	SDreaddata	sfrdata/ sfrcdata	Reads a slab of data from a data set.
	SDsetexternalfile	sfsextf	Defines the data type to be stored in an external file.
	SDwritedata	sfwdata/ sfwcdata	Writes a slab of data for a data set.
General Inquiry	SDfileinfo	sffinfo	Returns information about the contents of a file.
	SDgetinfo	sfginfo	Returns information about a data set.
	SDidtoref	sfid2ref	Returns a reference number for a named data set.
	SDiscoordvar	sfiscvar	Distinguishes data sets from dimension scales.
	SDnametoindex	sfn2index	Returns an index of a specified data set.
	SDreftoindex	sfref2index	Returns the index of a data set corresponding to a given reference number.

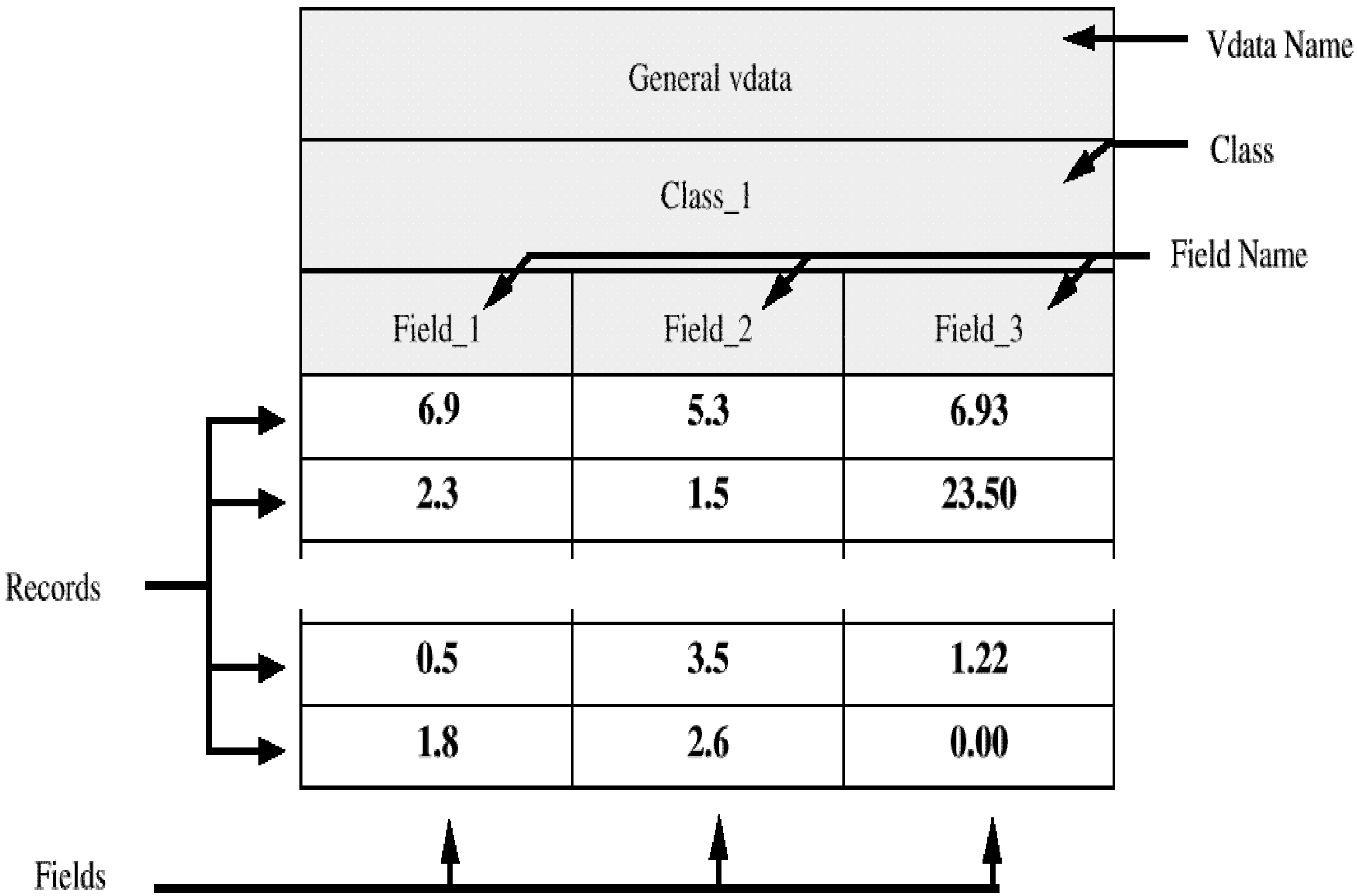
SDS Interface Routines (continued)

Dimension Scales	SDdiminfo	sfdinfo	Gets information about a dimension.
	SDgetdimid	sfdimid	Retrieves the identifier of a dimension.
	SDsetdimname	sfsdmname	Associates a name with a dimension.
	SDgetdimscale	sfgdscale	Returns scale values for a dimension.
	SDsetdimscale	sfsdscale	Defines the values of this dimension.
User-defined Attributes	SDattrinfo	sfgainfo	Gets information about an attribute.
	SDfindattr	sffattr	Returns the index of the specified attribute.
	SDreadattr	sfrnatt/ sfrcatt	Reads the values of the specified attribute.
	SDsetattr	sfsnatt/ sfscatt	Creates and defines a new attribute.
Predefined Attributes	SDgetcal	sfgcal	Returns calibration information.
	SDgetdatastrs	sfgdtstr	Returns the label, limit, format and coordinate system of a data set.
	SDgetdimstrs	sfgdmstr	Returns the attribute strings for a dimension.
	SDgetfillvalue	sfgfill/ sfgcfill	Reads the fill value if it exists.
	SDgetrange	sfgrange	Returns the range of values of the specified data set.
	SDsetcal	sfscal	Defines the calibration information.
	SDsetdatastrs	sfsdtstr	Defines the attribute strings of the specified data set.
	SDsetdimstrs	sfsdmstr	Defines the attribute strings of the specified dimension.
	SDsetfillvalue	sfsfill/ sfscfill	Defines the fill value of the current data set.
	SDsetrange	sfsrange	Defines the maximum and minimum values of the valid range.
Compression	SDsetcompress	None	Defines the compression method to be applied to data set data.

Vdatas (VS API)

- *The vdata object is a collection of records whose values are stored in fixed-length fields.*
- *The HDF **Vdata model** provides a framework for storing customized tables, or **vdatas**, in HDF files.*
- *The term "vdata" is an abbreviation of "vertex data" which alludes to the fact that, when the object was first implemented in HDF, it was designed specifically for the purpose of storing the vertex and edge information of polygon sets. The vdata design has since been generalized to apply to a broader variety of applications.*
- *Vdatas are uniquely identified by*
  - *a name,*
  - *a class*
  - *and a series of individual field names*

Vdata Table Example



Example of Different Possible Vdata Structures

Simulation Data 1		
2D_Temperature_Grid		
X	Y	Temp
2.30	1.50	23.50
3.40	5.70	8.03
0.50	3.50	1.22
1.80	2.60	0.00

3 Single-Component Fields

Simulation Data 1	
2D_Temperature_Grid	
X,Y	Temp
2.30, 1.50	23.50
3.40, 5.70	8.03
0.50, 3.50	1.22
1.80, 2.60	0.00

1 Multi-Component Field

1 Single-Component Field

Simulation Data 1
2D_Temperature_Grid
X, Y, Temp
2.30, 1.50, 23.50
3.40, 5.70, 8.03
0.50, 3.50, 1.22
1.80, 2.60, 0.00

1 Multi-Component Field

Interlaced and Non-Interlaced Vdata Contents

Vdata			
Mixed_Data_Type			
Temp	**Height**	**Speed**	**Ident**
1.11	**1**	**11.11**	**A**
2.22	**2**	**22.22**	**B**
3.33	**3**	**33.33**	**C**

File Interlacing: FULL_INTERLACE

Vdata			
Mixed_Data_Type			
Temp	**1.11**	**2.22**	**3.33**
Height	**1**	**2**	**3**
Speed	**11.11**	**22.22**	**33.33**
Ident	**A**	**B**	**C**

File Interlacing: NO_INTERLACE

Vdata Interface Routines

Category	Routine Name		Description
	C	Fortran-77	
Access	Vend	vfend	Closes the vdata interface.
	Vstart	vfstart	Initializes the vdata interface.
	VSattach	vsfatch	Establishes access to a specified vdata.
	VSdetach	vsfdtch	Terminates access to a specified vdata.
Read and Write	VHstoredata	vhfsd/vhfscd	Writes data to a simple, single-component vdata.
	VHstoredatam	vhfsdm/vhfscdm	Writes data to a simple, multi-component vdata.
	VSfdefine	vsffdef	Defines a new vdata field.
	VSsetclass	vstscls	Assigns a class to a vdata.
	VSsetfields	vsfsfld	Specifies the vdata fields to be written to.
	VSsetinterlace	vsfsint	Sets the interlace mode for a vdata.
	VSsetname	vsfsnam	Assigns a name to a vdata.
	VSwrite	vsfwrt/vsfwrtc	Writes records to a vdata.
	VSread	vsfrd/vsfrdc	Reads from a vdata.
	VSseek	vsfseek	Seeks to a specified record in a vdata.
	VSfpack	vsfnpak/vsfcpak	Packs field data into a buffer or unpacks field data from a buffer.
File Inquiry	VSfind	vsffnd	Searches for a given vdata name the opened HDF file.
	VSgetid	vsfgid	Returns the identifier of the next vdata in the file.
	VSlone	vsflone	Returns the vdatas that are not linked into vgroups.

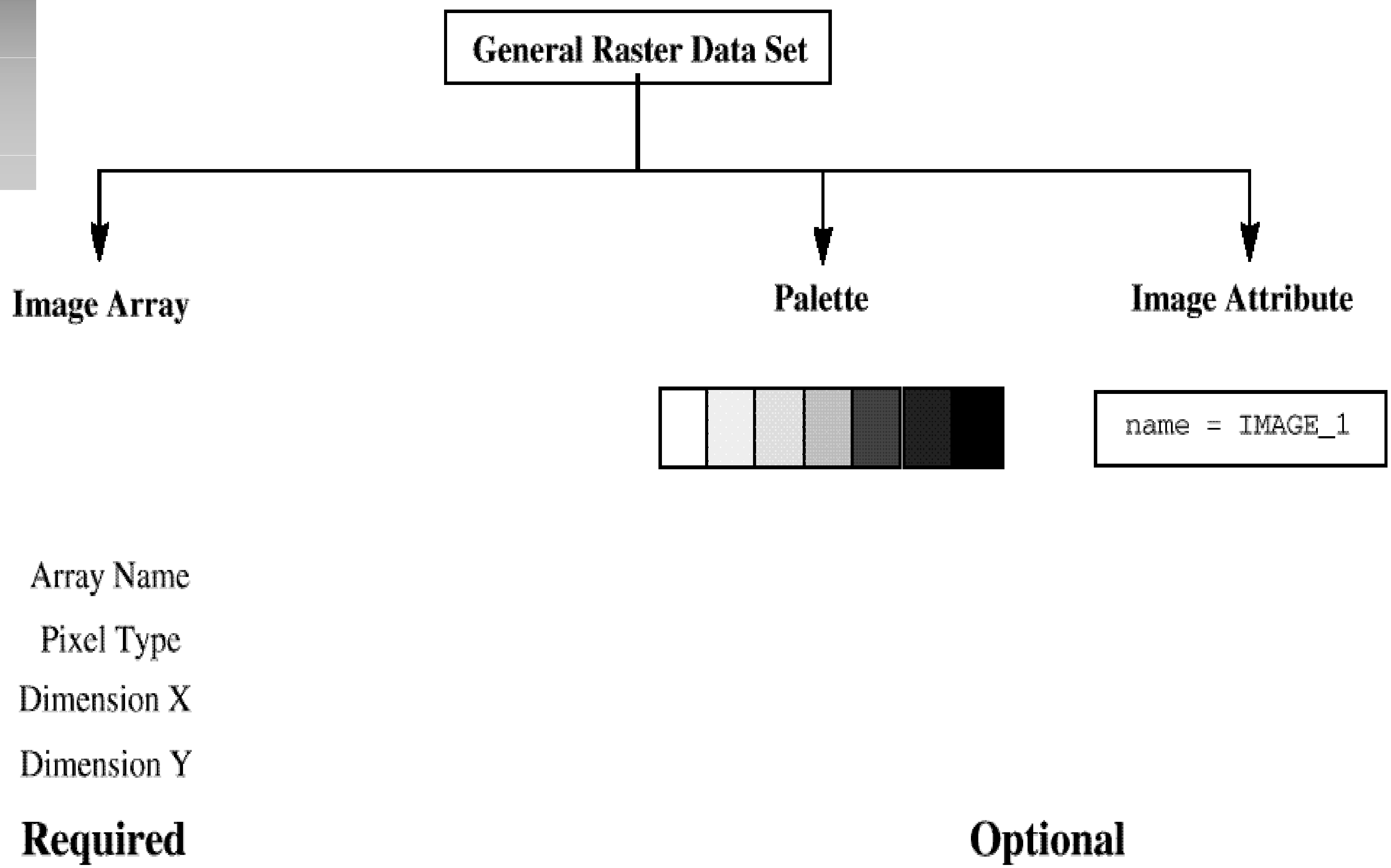
Vdata Interface Routines (continued)

Vdata Inquiry	VSfexist	vsfex	Tests for the existence of fields in the specified vdata.
	VSinquire	vsfinq	Returns information about the specified vdata.
	VSelts	vsfelts	Returns the number of records in the specified vdata.
	VSgetclass	vsfcls	Returns the class name of the specified vdata.
	VSgetfields	vsfgfld	Returns all field names within the specified vdata.
	VSgetinterlace	vsfgint	Retrieves the interlace mode of the specified vdata.
	VSgetname	vsfgnam	Retrieves the name of the specified vdata.
	VSsizeof	vsfsiz	Returns the field sizes of the specified vdata.
	VSQueryclass	None	Returns the class of the specified vdata.
	VSQueryfields	vsgfld	Returns the field names of the specified vdata.
	VSQueryname	vsfgname	Returns the name of the specified vdata.
	VSQueryref	None	Retrieves the reference number of the specified vdata.
	VSQuerytag	None	Retrieves the tag of the specified vdata.
	VSQuerycount	vsfelts	Returns the number of records in the specified vdata.
	VSQueryinterlace	vsfgint	Returns the interlace mode of the specified vdata.
	VSQueryvsize	vsfsiz	Retrieves the local size in bytes of the specified vdata record.
Field Inquiry	VFfieldesize	None	Retrieve the field size (as stored in a file) of a specified field.
	VFfieldisize	None	Retrieve the field size (as stored in memory) of a specified field.
	VFfieldname	None	Retrieves the name of the specified field in the given vdata.
	VFfieldorder	None	Retrieves the order of the specified field in the given vdata.
	VFfieldtype	None	Retrieves the data type for the specified field in the given vdata.
	VFnfields	None	Retrieves the total number of fields in the specified vdata.

General Raster Images (GR API)

- *HDF users familiar with the SD interface will find the general raster data model a simplified version of the SD scientific data set model, customized to accommodate image data storage and manipulation.*
- *The raster image data is stored in a two-dimensional array and attributes can be created for the image, the file or both.*
- *Palettes can be created and attached to the image as well as compression method information.*
- *A fundamental difference between the SD scientific data model and the GR raster data model is the absence of customizable dimensional information in the GR data set.*

General Raster Image Data Set Contents



GR Interface Routines

Purpose	Routine Name		Description
	C	Fortran-77	
Access	GRstart	mgstart	Initializes the GR interface for a given data file.
	GRend	mgend	Terminates access to the file initialized by **GRstart**.
	GRselect	mgselct	Selects the data set to perform operations on.
	GRendaccess	mgendac	Terminates access to the data set selected by **GRselect** or **GRcreate**.
Read/write	GRreadimage	mgrdimg/ mgrcimg	Reads image data from a general raster data set.
	GRwriteimage	mgwrimg/ mgwcimg	Writes image data to a general raster data set.
	GRidtoref	mgid2ref	Maps an general raster data set identifier to a reference number.
	GRreftoindex	mgr2idx	Maps the reference number of a general raster data set to a data set index and returns the index.
	GRnametoindex	mgn2ndx	Maps a raster image name to an index and returns the index.
	GRreadlut	mgrdlut/ mgrclut	Reads palette data from a general raster data set.
	GRwritelut	mgwrlut/ mgwclut	Writes palette data to a general raster data set.
	GRsetattr	mgsnatt/ mgscatt	Writes the attribute of an object to a general raster data set.
	GRgetattr	mggnatt/ mggcatt	Reads the attribute of an object from a general raster data set.

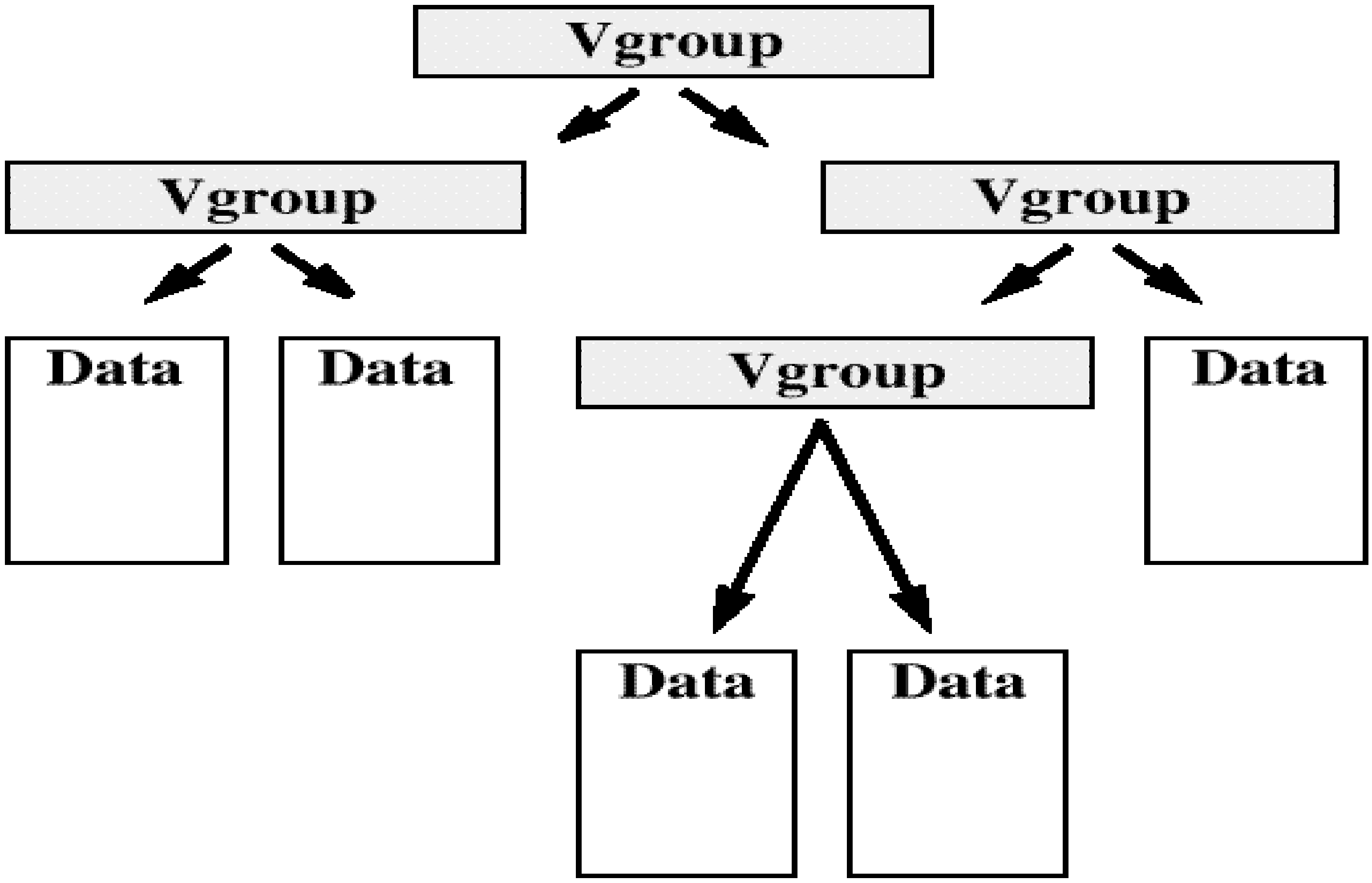
GR Interface Routines (continued)

Maintenance	GRcreate	mgcreat	Creates a new general raster data set.
	GRreqlutil	mgrltil	Sets the interlace mode for the next palette read from a general raster data set.
	GRreqimageil	mgrimil	Sets the interlace mode for the next image read from a general raster data set.
	GRgetlutid	mggltid	Allocates a palette id to a general raster data set.
	GRsetexternalfile	mgsxfil	Specifies that the image data of a general raster data set is a special element of an external element.
	GRsetaccesstype	mgsactp	Sets the access to a general raster data set to be either parallel or serial.
	GRsetcompress	mgscomp	Makes the image data of a general raster data set a compressed special element.
Inquiry	GRfileinfo	mgfinfo	Returns global information on the specified GR data set.
	GRgetiminfo	mggiint	Returns information on the specified general raster data set.
	GRgetlutinfo	mgglinf	Returns information on a given palette.
	GRattrinfo	mgatinf	Returns attribute information on a given object.
	GRfindattr	mgfndat	Returns the index of an attribute with the given name.

Vgroups (V API)

- *A **vgroup** is a structure designed to associate related objects.*
- *Data organization within a vgroup resembles the UNIX file system.*
- *The general structure of vgroups are similar to UNIX directories or subdirectories in that a vgroup may contain references to other vgroups or data objects.*
- *In previous versions of HDF, the data objects in a vgroup were limited to vdatas. Any HDF data object can now be included within a vgroup.*

Vgroup Example



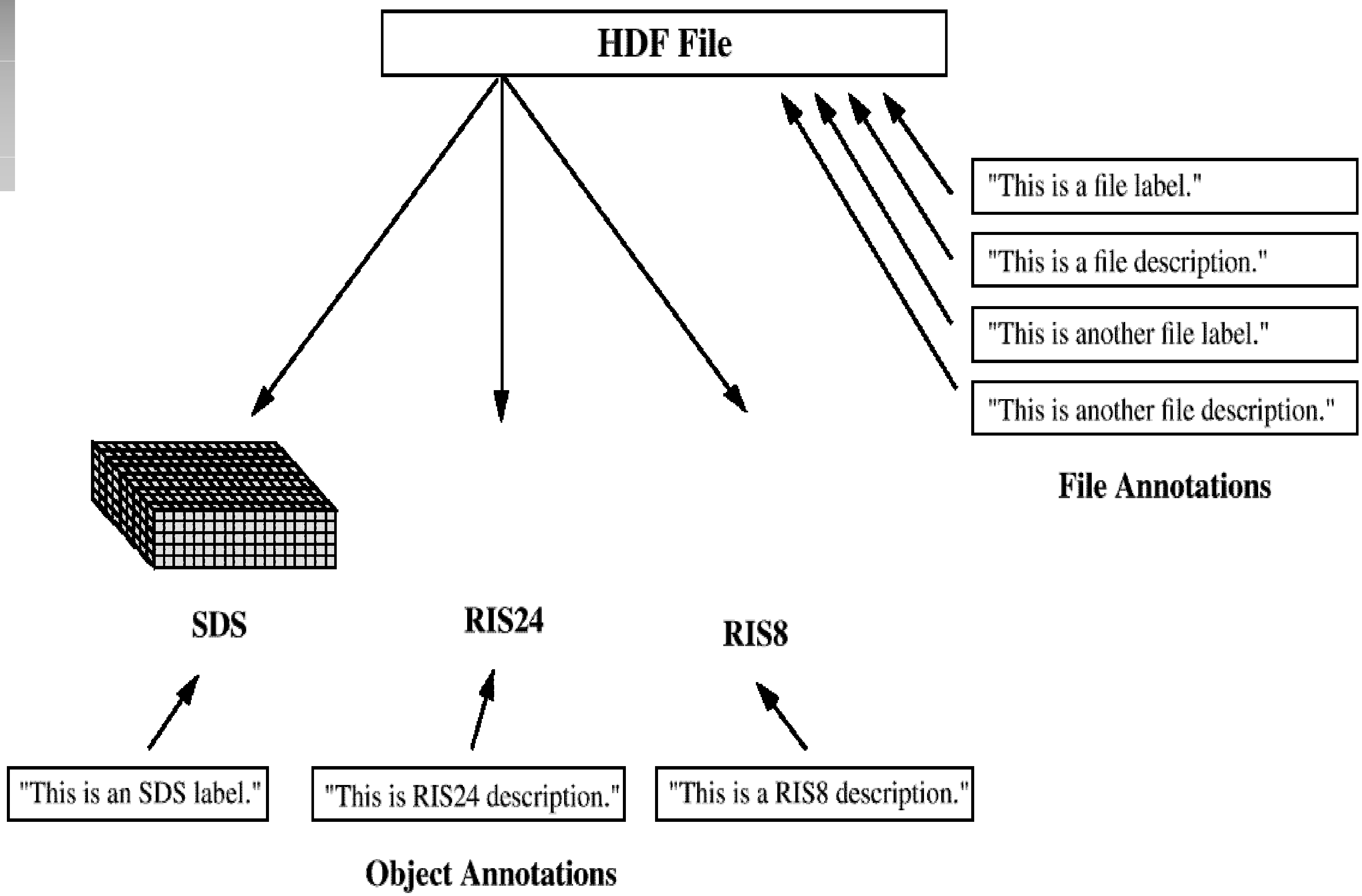
Vgroup Interface Routines

Category	Routine Name		Description
	C	Fortran-77	
Access	Hopen	hopen	Opens the specified HDF file.
	Hclose	hclose	Closes the specified HDF file.
	Vstart	vfstart	Initializes the V interface.
	Vattach	vfatch	Establishes access to a vgroup.
	Vdetach	vsfdtch	Terminates access to a vgroup.
	Vend	vfend	Terminates access to the V interface.
Create	Vsetclass	vfscls	Assigns a class to a vgroup.
	Vsetname	vfsnam	Assigns a name to a vgroup.
	Vinsert	vfinsrt	Adds a vgroup or vdata to an existing vgroup.
	Vaddtagref	vfadtr	Adds any HDF data object to an existing vgroup.
File Inquiry	Vlone	vflone	Returns the reference numbers of vgroups not included in other vgroups.
	Vgetid	vfgid	Returns the reference number for the next vgroup in the HDF file.
Vgroup Inquiry	Vinquire	vfinq	Returns general information about a vgroup.
	Vgetclass	vfgcls	Returns the class of the specified vgroup.
	Vgetname	vfgnam	Returns the name of the specified vgroup.
	Visvg	vfisvg	Checks if a vgroup identifier belongs to a vgroup within a vgroup.
	Visvs	vfisvs	Checks if a vdata identifier belongs to a vdata within a vgroup.
	Vgettagref	vfgttr	Retrieves a tag/ reference number pair for a data object in the specified vgroup.
	Vntagrefs	vfntr	Returns the number of tag/reference number pairs contained in the specified vgroup.
	Vgetnext	vfgnxt	Returns the identifier of the next vgroup or vdata in a vgroup.
	Vgettagrefs	vfgttrs	Retrieves the tag/reference pairs of all of the data objects with a vgroup.
	Vinqtagref	vfingtr	Checks if an object belongs to a vgroup.

Annotations (DFAN and AN API)

- *When working with different data types, it is often convenient to identify the contents of a file by adding a short text description or **annotation**.*
- *An annotation serves as the label for a file or data element.*
- *As they are implemented as variable-length strings, HDF annotations are designed to accommodate a wide variety of information including titles, comments, variable names, parameters, formulas, and source code.*
- *In fact, HDF annotations can encompass any textual information regarding the collection, meaning, or intended use of the data.*

File and Object Annotations



DFAN Interface Routines

Purpose	Functions		Description
	C	Fortran-77	
Write	DFANaddfid	daafid	Assigns a file label to a specific file.
	DFANaddfds	daafds	Assigns a file description to a specific file.
	DFANputlabel	daplab	Assigns an object label to a specific data object.
	DFANputdesc	dapdesc	Assigns an object description to a specific data object.
Read	DFANgetfidlen	dagfidl	Returns the length of a file label.
	DFANgetfid	dagfid	Reads the text of a file label.
	DFANgetfdslen	dagfdsl	Returns the length of a file description.
	DFANgetfds	dagfds	Reads the text of a file description.
	DFANgetlablen	dagllen	Returns the length of an object label.
	DFANgetlabel	daglab	Reads the text of an object label.
	DFANgetdesclen	dagdlen	Returns the length of an object description.
	DFANgetdesc	dagdesc	Reads the text of an object description.
General Inquiry	DFANlablist	dallist	Gets a list of all the labels in a file for a particular tag.
	DFANlastref	dalref	Returns the reference number of the last annotation accessed.

Limitations

- *H-Level Limits*
  - *files open at a single time: 32*
  - *access records open at a single time: 256*
  - *file size: 1 GB (?)*
- *Vdata Limits*
  - *fields in a Vdata: 256*
- *SD Limits*
  - *attributes for a given object: 3000*
  - *maximum dimensions per data set: 5000*
  - *maximum variables per data set: 5000*
  - *maximum per variable dimensions: 32*

HDF Version 5 Preview

- *no file size restrictions*
- *improved performance*
- *parallel input/output*
  - *MPI*
  - *network-of-workstations*
- *object-oriented*
- *backward-compatible*
- *netCDF-compatible*

Public Domain HDF Software

- NCSA HDF Utilities
  - conversion utilities
  - tools for analyzing the contents of an HDF file
  - tools for manipulating HDF files
- NCSA Java-based HDF Viewer
  - an interactive tool for viewing an HDF file
- NCSA X DataSlice
  - allows the user to manipulate 3D images under X11, using the HDF file format and libraries
- DDI
  - extracts only the relevant data and providing it to a chosen graphics engine in the required form without undue effort
  - reads and writes a number of publicly available file formats
  - sends data to public domain and commercial visualization systems

Public Domain HDF Software (continued)

- *Envision*
  - *an interactive system for the management and visualization of large scientific data sets*
  - *runs under X/Motif*
  - *manages data stored in HDF or netCDF files*
  - *does visualization using IDL, NCSA Collage, and NCSA XDataSlice*
- *GRASS*
  - *an integrated set of programs designed to provide digitizing, image processing, map production, and geographic information system capabilities to its users*
- *HDF Browser*
  - *for Windows and Macintosh*
  - *offers point-and-click access to data stored in the HDF format*

Public Domain HDF Software (continued)

- *HDFLook*
  - *Motif HDF viewer, useful for quality control of Scientific Data Sets*
  - *allows easy access to physical values and ancillary data, and includes 2-D graphics (radial, histogram)*
- *LinkWinds*
  - *visual data analysis and exploration system designed to rapidly and interactively investigate large multivariate and multidisciplinary data sets to detect trends, correlations and anomalies*
- *Ingrid*
  - *designed to manipulate large data sets and model input/output*
  - *reads and writes netCDF files, writes HDF files*
  - *generates plots, including line, contour, vector, and scatter plots, as well as histograms*

Public Domain HDF Software (continued)

- *SciAn*
  - *scientific visualization and animation package for Silicon Graphics workstations*
  - *brings together the power of 3-dimensional scientific visualization and movie making with the ease of use and familiarity of object-oriented drawing packages*
- *VCS*
  - *facilitates the selection, manipulation, and display of scientific data*
  - *user gains virtually complete control over the appearance of the data display and associated text*
- *VISTAS*
  - *an interactive, large volume data browsing and probing environment*

Commercial HDF Software

- *AVS*
  - *incorporates latest advances in visualization software, graphics, networking, high performance systems, and industry standards into a single comprehensive visualization environment*
  - *offers data input and output modules which will read and write files in HDF format*
- *Data Explorer*
  - *general-purpose software package for data visualization and analysis*
- *ER Mapper*
  - *an advanced digital image processing and remote sensing system created to help earth scientists integrate, enhance, visualize, and interpret their geographic data*
  - *allows truly interactive “real time” integration and processing of data*

Commercial HDF Software (continued)

- *IDL*
  - *software package for data analysis, visualization, and application development*
  - *advanced image processing, interactive 2D and 3D graphics, insightful volume visualization, a high-level programming language, integrated mathematics and statistics, flexible data I/O, a cross-platform GUI toolkit, and versatile program linking tools*
- *IRIS Explorer*
  - *powerful yet easy-to-use sophisticated visualization system with a user environment that allows users and application developers to build complex applications for visualizing sets of data.*

Commercial HDF Software (continued)

- *Noesys*
  - *opens HDF files, displays their full contents, and provides editors for working with all of the types of scientific data that can be stored within an HDF file*
  - *includes a powerful Fortran-based data manipulation language, along with easy-to-use visual data analysis tools, Plot, Transform, and T3D, for menu driven plotting, rendering, and image analysis*
  - *can import any ASCII and binary data, create annotations, macros, images and color palettes specific to the data and save it all as an HDF file*
- *PV-Wave*
  - *for solving problems requiring the application of graphics, mathematics, numerics, and statistics to data and equations.*
- *MATLAB*

Web Site — http://hdf.ncsa.uiuc.edu/

TRANSIMS Data Requirements

- *metadata*[†]
- *variety of data formats*
  - *tabular/relational*[*]
  - *multidimensional arrays*[*]
  - *bitmaps (?)*[*]
- *large data sets*[‡]
- *indexing*
- *sorting*[¶]
- *filtering*[¶]
- *compression*[†]
- *parallelism*[‡]
- *high-performance*[‡]
- *platform-independence*[*]

[*] supported fully by HDF4
[†] supported partially by HDF4
[‡] to be supported by HDF5
[¶] supported by HDF-compatible analysis software

TRANSIMS Data Files

- *database subsystem files*
  - *network data tables*
  - *microsimulation output specifications*
- *plan files*
  - *planner output files (text)*
  - *post-processed plan files (binary) for input to CA microsimulation*
- *simulation output subsystem files*
  - *evolution/trajectory data*
  - *event data*
  - *summary data*
- *post-processed simulation output*
  - *various ad-hoc formats*

Possible uses of HDF in TRANSIMS

- a platform-independent file import/export format
- a bridge to numerous data analysis and visualization software packages
- a native format for all TRANSIMS data

Future Plans for HDF in TRANSIMS

- *develop HDF connectivity to current TRANSIMS files*
  - *import/export utility for the TRANSIMS database subsystem*
  - *export utility for the TRANSIMS simulation output subsystem*
- *evaluate HDF-compatible data analysis and visualization software*
- *monitor progress on HDF5 and consider it as a candidate for the native TRANSIMS data format in the future*
  - *parallel input/output needed*
  - *support for very large data sets needed*
  - *C++ and Java interfaces needed*